ARMY RESEARCH LABORATORY

A Discussion of Metrics for Parallelized Army Mobile ad hoc Network Simulations

by Dale Shires, Kenneth Renard, and Brian Henz

ARL-MR-0792 September 2011

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

Army Research Laboratory
Aberdeen Proving Ground, MD 21005

ARL-MR-0792 **September 2011**

A Discussion of Metrics for Parallelized Army Mobile ad hoc Network Simulations

Dale Shires, Kenneth Renard, and Brian Henz
Computational and Information Sciences Directorate, ARL

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)* September 2011	**2. REPORT TYPE**	**3. DATES COVERED (From - To)**

4. TITLE AND SUBTITLE A Discussion of Metrics for Parallelized Army Mobile ad hoc Network Simulations	**5a. CONTRACT NUMBER**
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Dale Shires, Kenneth Renard, and Brian Henz	**5d. PROJECT NUMBER**
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) U.S. Army Research Laboratory ATTN: RDRL-CIH-C Aberdeen Proving Ground, MD 21005	**8. PERFORMING ORGANIZATION REPORT NUMBER** ARL-MR-0792
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	**10. SPONSOR/MONITOR'S ACRONYM(S)**
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Mobile *ad hoc* networks will provide the increasingly net-centric Army fighting force with the ability to communicate across a highly dynamic and evolving operational space. While these networks share traits of commercial wireless networks, they are unique in many regards. Highly mobile forces using communication devices with varying signal strengths and capacities create networks that are difficult to characterize, control, and optimize. There are several approaches that researchers can use to mitigate these challenges, the topic here being network simulation. Network simulation is a valuable tool to model network behavior and study the interactions of digitized devices within the Army battlespace. Given the large size of deployed Army networks, it is desirable to have scalable network simulators where large networks can be studied and optimized in synthetic environments. This report discusses network simulators at a high level within the context of Army mobile *ad hoc* networks. Further, it discusses key metrics that we feel need to be addressed to determine viability and utility to mobile *ad hoc* network centric warfare simulators.

15. SUBJECT TERMS

Simulation, parallel discrete event simulation

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Dale Shires
a. REPORT Unclassified	**b. ABSTRACT** Unclassified	**c. THIS PAGE** Unclassified	UU	24	**19b. TELEPHONE NUMBER** *(Include area code)* (410) 278-5006

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

Contents

List of Figures

Acknowledgments

This work was supported in part by a grant of computer time and resources by the Department of Defense High Performance Computing Modernization Program as part of the Mobile Network Modeling Institute.

1. Introduction

The U.S. Army is becoming an increasingly net-centric warfighting organization. Digital networks of Internet Protocol (IP) devices scaling to large numbers of fixed and mobile devices are becoming the norm for Army deployed operations in all parts of the world. These networks have the potential to allow for increased capabilities and situational awareness as sensors, unmanned craft, and Army warfighters are equipped with an array of communication and processing devices each possessing a unique IP address. However, without proper planning and optimization, this larger array of devices can quickly become a fragile communication infrastructure with high failure rates.

Wireless networks of the modern world have provided almost instantaneous access to voice communications and the World Wide Web from academic campuses to coffee houses to individual homes. However, shortcomings are all too obvious: dropped calls, variations in data communication rates based on distances from network hubs, interference from obstacles and other networked devices, etc. Individuals will often move to other areas with less interference to overcome these issues. Service providers will compensate by adding increased access points in well-known troubled areas.

The bulk of the research being conducted in academia is focused on wireless cellular networking as described above. While aspects of cellular networks carry through to mobile networks, the two are distinct in many ways with mobile networks posing far greater challenges. In mobile networks, access points can move and coverage may vary widely in a region. At times, the access points will cluster together, leaving parts of the map with sparse coverage and parts with compromised service due to competition for available bandwidth as channel subscriptions become overwhelmed. On the battlefield, relocating receivers to areas with better coverage may not be an option. Consider the problem of combat in an urban area. Narrow streets and buildings with metal roofs and reinforced concrete walls may interfere with radio reception and access to surveillance data, yet that surveillance information may be key to locating enemy combatants before they inflict casualties on friendly forces.

Army mobile ad hoc networks (MANETs) also pose unique challenges due to the potentially large scale at which they may operate; high throughput requirements for video, voice, and data traffic; and less than friendly operational environments that can lead to signal decay. Wireless devices within this network may consist of vehicle-mounted radios, dismounted Soldiers with various radios, sensor systems, unmanned aerial vehicles, space-based systems, etc. These mobile networks must be understood at a variety of levels. In terms of the Open Systems Interconnection model (OSI model), this can range from low-level physical layers all the way up the stack to the application layer. Much of this determination of importance gets to the heart of the question at hand. If we are concerned only with a question of can two devices “see” each other, much of this can be addressed at the physical layer with fidelity from terrain and weather factoring. Impact on performance from, say, a security application riding on the network, may have to be addressed at the application layer.

There are many agencies, groups, and individuals who can benefit from a better understanding of how to design, deploy, and operate a mobile network. Army and Department of Defense (DoD) commanders need to understand how to deploy forces to allow unfettered communications that will support real-time processing in situations calling for command and control interaction. Mobile device designers need to understand how their devices should be configured in a wider networked environment to determine the right decisions that will impact latency and bandwidth, and balance those with operational issues such as battery power. In situations where models might be incomplete, the ability to factor in past performance and experimental data will be invaluable to fill data voids.

With this in mind, the U.S. Army Research Laboratory (ARL) and its partners have started an effort known as the Mobile Network Modeling Institute (MNMI) with funding and support provided by the DoD High Performance Computing Modernization Program Office. In order to address as many issues as possible in MANET planning and optimization, the MNMI is addressing the topic in four overlapping areas; network emulation, network simulation, network experimentation, and the Network Interdisciplinary Computing Environment (NICE). NICE is an environment allowing for commonality across the components and also supports scientific visualization with hooks that will allow for analysis (data mining and analytics). Network emulation, experimentation, and NICE are covered in detail in separate documents (*1*). The focus of this monograph is network simulation and the discussion of what metrics are important to the topic from an Army warfighter perspective.

2. Network Simulation

Network simulation is one component of the MNMI that will assist researchers and developers with the tools to help design and optimize networks before they are deployed and as they evolve during operation. In the context of Army operations, network simulations can be helpful to developers and designers of new communication protocols. Take, for example, the development of a new radio or even an unattended ground sensor (UGS). These devices can easily scale to several hundred or even several thousands of deployed nodes. How well a proposed protocol might scale to these levels is an important consideration. Given the fact that the Army has to deploy to any number of different environmental settings and conditions, network simulation can be used to test these protocols at various scales and configurations in parametric studies to hone key parameters.

Wireless ad hoc networks pose significant and different challenges over their wired counterparts (*2*). Some of these challenges include the following:

- Non-stationary nodes that can cluster in some areas and result in lack of coverage in others
- Dynamic signal interference and attenuation from other networked nodes or from terrain/weather effects
- Broadcast routing that can create dynamic network overload
- Realistic traffic generation

All of these areas have to be addressed within the context of mobile networks for the simulation engine to generate meaningful results. In some cases, such as determining RF propagation and the physical layers, MANETs pose unique challenges over traditional wired networks. Indeed, it is the complexity of these MANETs that invalidates the use of analytical methods for evaluating the scalability of protocols (*3*). Accordingly, simulations are most often the best fit and most often employed techniques to assess MANETs.

2.1 Discrete Event Simulation

Several network simulators have been developed over the years including Yet Another Network Simulator (YANS), ns-2 and ns-3, OPNET, and ROSS (*4–8*). All of these approaches employ a discrete event simulator (DES). A DES maintains the global state of a system while events associated with a clock are executed. Events are processed from an event queue and the effects of the event are determined and propagated throughout the system.

Naturally, this should model the system correctly as long as the DES processes the events correctly and in time sequence. However, this poses a problem for large-scale simulations. It is true that the large-scale memories available in high performance computing (HPC) systems will allow for truly large network simulations simply because enough devices can be maintained in global or distributed memories. However, it is not difficult to imagine how poorly such a system will perform if the DES operates strictly sequentially. Further complicating this is the broadcast-like nature of wireless networks in MANETs. Node transmissions can cause a cascade of events to include events representing arrival of transmissions at all nodes within range. Parallel DESs have been the topic of research to help mitigate these scaling problems (*3*).

2.2 Parallel Discrete Event Simulation

Parallel discrete event simulation (PDES) has the potential to alleviate difficulties associated with simulating large-scale networks by both adding more memory to the simulation and also providing the capability for non-interference events to proceed concurrently. That is, in this environment, there is no global clock to slow the system down. The system being modeled consists of physical processes that interact over time and logical processes, one per each physical process, that interact over time (*9*). In the case of Army MANETs, these physical processes may be various radios or UGSs. Logical processes communicate with each other by sending time-stamped events.

Systems such as this pose the potential problem of introducing causality errors in the simulation if nothing is done to maintain order (*9*). That is, imagine the state variables for logical process LP_1 are updated at time T_8 on the global simulation clock. This logical process might be mapped to processor 243 on a 1,024 processor HPC system. This update causes LP_1 to generate a communication message at time T_9. Put in the context of an Army MANET simulation, it is quite possible that at global simulation time T_3 on a different processor, say processor 122, an event could have happened causing all communications from LP_1 to stop. The entire system is now unbalanced and incorrect.

In order to deal with this, researchers have developed three approaches to avoid or correct causality errors (*10*). The first approach, conservative, does not allow causality errors to enter the system. An LP does not execute an event until it is certain that no other events with an earlier timestamp will arrive at that LP. The second method, optimistic, avoids dead waiting time in the simulation by letting the LPs process events "optimistically." If an event arrives later that is out of order, the LP must compensate somehow to prior states. Reverse computation is one way to perform this unrolling of events (*11*). It is safe to assume that there might be situations where a third mixed-mode approach using both of these may be useful.

3. Critical Metrics

In this section, we address several critical metrics common with network simulation and also comment on factors that only reveal themselves when discussing large-scale concerns that become obvious with large MANETs, as will be the case with Army networks. Some of these metrics are quantitative; they can be assessed through a well-defined equation. At the same time, many of these are qualitative; gauging success or failure is more up to the user and all of the history that person might bring to the table. In either case, this list is not meant to be exhaustive but rather enumerates those we feel to be most important.

3.1 Performance

In terms of performance, parallel network simulators will often be most reliant on efficient and scalable parallel discrete event simulations. Wall clock time required to complete a simulation event is our primary metric (*12*). Assuming that the simulation parameters are identical across the various network simulators being studied, the one with the highest throughput of events is the most favorable. This comes with a subtle caveat. For optimistic simulators, a high event throughput would be expected, but rollback for causality may, at times, largely undo events very quickly. For this reason, we stress fixed network simulation times with identical parameters to fairly judge performance. Some level of parametric study should be employed with varying parameters to assess network simulators in different scenarios (such as drop probability).

Somewhat complicating this metric is the type of parallel computer that will be used. Since HPC hardware can take many forms, a type of network simulator optimized for one system might not perform so well on another. Various parallel methodologies exist, such as message passing or loop parallelism, and in fact, there are other numerous evolving architectures, i.e., those employing accelerators such as general purpose graphics processing units (GPUs). An adequately detailed description of the underlying hardware assumptions made by a network simulator is key in order to understand a priori the type of performance that might be expected on certain hardware configurations.

Memory usage is another key metric for network simulators, and for large-scale systems, in particular. Effective use of memory will allow for larger simulations at every level. Problems with memory, such as memory leaks, become quite evident at the scales of interest here and can cause simulators to fail quickly.

3.2 Scalability

There are actually several meanings of the term "scalability" and we intend to use several of them here. To begin with, scalability can refer to fixed problem size speedup with increasing numbers of processors. This type of scaling is also known as strong scaling. In this regard, the number of processors used to perform a simulation is increased while the actual problem being studied remains fixed. (Scaled speedup, where problem sizes vary with the number of processors, is not discussed here since it is not a typical concern within the simulation problem domain.) Linear speedup is considered the best one can do, with superlinear speedup often, and rightly, explained away by variances in behavior from the performance of memory systems. An example is illustrative. Say, for instance, that a simulation takes 512 s on one processor. If we add 511 processors and the simulation now takes 1 s, we have achieved linear speedup. If the simulation only takes 0.5 s, then we have achieved superlinear speedup as we are completing faster than should be possible.

Scalability is linked with performance in that achieving good scalability should, in most cases, lead to enhanced performance. One processor simulations, also known as sequential simulations, usually form the baseline of scalability and performance assessment. However, the algorithm that performs best on one processor may not necessarily be the same algorithm that is designed for parallelism. A good example is merge sort. On multiple processors, merge sort might be a good option. While this algorithm also works on only one processor, there might be other better alternatives, such as quick sort. In looking at scalability, if we use the same parallel algorithm to evaluate one processor execution times, that speedup is referred to as *relative speedup*. If, instead, we use the best sequential algorithm as a baseline, that speedup is known as *absolute speedup* (*13*). For HPC systems, relative speedup is usually of the most concern since we are interested in scaling performance with numerous processors.

The actual performance achieved is also subject to Amdahl's Law. Roughly speaking, Amdahl's Law is given as

$$T_{total} = T_{serial} + T_{parallel} \tag{1}$$

where T is the run time. The total run time consists of the run times for the sequential, or serial one processor portion of the algorithm, plus the time required for those parts of the algorithm that can run in parallel. As we add more and more processors, the total execution time becomes dominated by the time that we cannot improve in parallel. Amdahl's Law in terms of speedup is given as

$$Speedup = \frac{s+p}{s+\frac{p}{n}} \tag{2}$$

where s is the amount of time spent in the serial portion of the code, p is the amount of time spent in the parallel portion of the code, and n is the number of processors.

How this can quickly become a problem is highlighted in figure 1. This is a hypothetical scenario of a code that is 50% serial, 50% parallel, and takes one hour wall clock time to execute on one processor. On two processors, we are stuck with 1800 s of serial time regardless of the parallel portion, that can now run in 900 s. We can continue to cut the run time of 50% of the code, but are stuck with 1800 s that we can never get rid of. This highlights the importance of good parallel algorithm development in trying to remove serial bottlenecks as much as possible.

It should be noted that superlinear performance is not that uncommon. Largely this is due to overtaxed memory systems when using only one processor on an HPC system and a fixed problem size that is "relatively" large. Cache behavior leading to translation lookaside buffer (TLB) misses or other spill code may lead to long run times that are mitigated by going to larger processor pools. Problem sizes usually decrease in such settings as the data sizes are decomposed across multiple processors. This leads to more

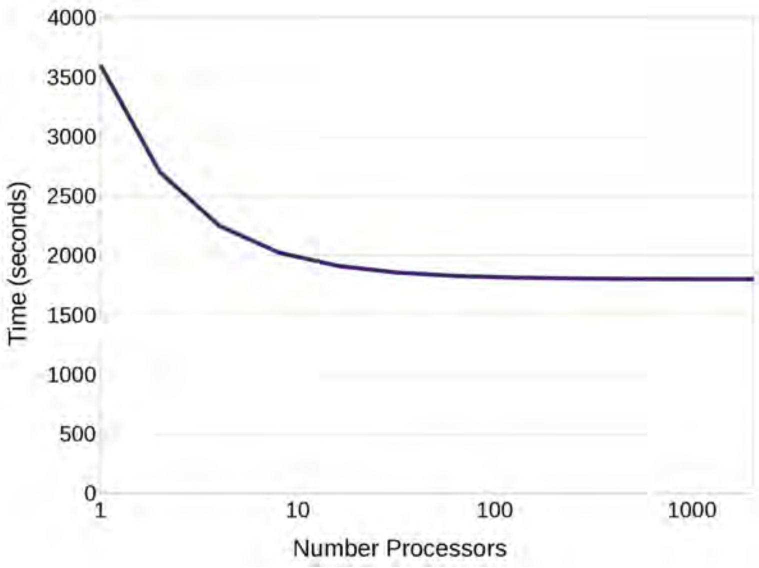

Figure 1. Limits on speedup with 50% of code being serial.

efficient code, and hence, what could be considered an unfair comparison. It is therefore not uncommon to relate more to a relative speedup that seeks a different baseline. Often this means trying to find the lowest processor count configuration where memory effects are sufficiently low to provide a good base processor count rather than simply using one processor as the start point (*14*). Doing so makes the speedup metric more meaningful.

3.3 Model Fidelity

Performance metrics have little meaning without consideration of the fidelity of the models. A simulation that does not implement all necessary parts of a protocol or estimates a computation instead of completing the full series of calculations cannot be considered identical simulations. For example, the performance of a User Datagram Protocol (UDP) model that does not perform checksum verification should not be compared against another model that does perform the verification without special consideration for its effect. While optimization techniques such as these that ignore small computations or entire protocol modeling may be completely sufficient for some problem sets, performance comparisons must be done on comparable levels of model fidelity.

Model complexity must also be considered in the same way to realize that higher fidelity models are likely to be more complex and models may have additional features that contribute to increased complexity but not necessarily to performance.

3.4 Ease of Use

Setting up large-scale network simulations is not a trivial task. Asset placement, description, and mobility all have to be defined. How this is accomplished in large part determines overall ease of use. Descriptor files, such as plain text files, can be used. Efforts to provide the tools and graphical user interfaces (GUIs) for application codes can easily take up much of the time in pushing utilities to users.

Recent efforts have been made by those in the Army communications laboratories to field technologies that will help in several areas. These include simulation setup, generating realistic communications effects, and something else. The Communications Planner for Operational and Simulation Effects with Realism (COMPOSER) is one such system (*15*). It consists of a network planner, network visualizer, and communications environmental effects module.

As part of the MNMI, researchers are establishing the NICE to facilitate the sharing of network data within simulation, emulation, and live experimentation events. Part of this methodology includes the Network Data Model and Format (NetDMF). Recent efforts have focused on adding NetDMF awareness to the COMPOSER toolkit.

Commercial tools, such as OPNET, also maintain a suite of tools for analysis of network simulation results.

Network simulators that are able to interface to these systems can be a big help for Army MANET researchers. Leveraging these components, which are built as part of a more open framework over commercial approaches, usually leads to a concentration of effort on other factors, such as performance, and less on reinventing the wheel.

3.5 External Interfaces

MANET simulation holds the potential to enhance emulation and experimentation by interfacing directly with “real world” systems. Network events within a PDES do not necessarily contain the required information to communicate with real systems external to the simulation. Examples of an implementation of this capability in simulators is through real-time execution in ns-3 and Qualnet (EXata). There are many potential advantages that this feature can provide, including access to advanced radio models and increased numbers of perceived radios in an experiment. The interactions between simulated and real or emulated devices can be at a one to one ratio or a many to one. In a one to one scenario, each simulated device is connected directly to the experiment and can communicate directly to any other device in the experiment. With a many to one ratio, the number of simulated devices can be quite large but only one or a few direct paths into the experiment are available. Each method has advantages but both require a simulator to be capable of executing in real time.

4. Miscellaneous Metrics

Scalability, performance, and ease of use dominate our criteria for optimal large-scale network simulators. However, there are other issues that should not be ignored in the development of these systems or their assessment for applicability.

4.1 Network Simulation Metrics

Discrete event simulations have their own set of performance characteristics, typically centered around the measurement of events computed over time. Models for network simulation can be implemented in various ways that can have different event counts for the same logical operation. A normalization of the event statistic is necessary when comparing performance of network simulation. An example of a model-agnostic event metric might be a "packet received event" that should be consistent across model implementations.

4.2 Intellectual Property Rights

The Army's unique requirements for MANET simulation poses the risk that only a limited commercial marketplace will be able to address requirements in network simulation tools that might be offered. A lack of competition and over-reliance on one approach has the tendency to stifle creativity and innovation and raise costs. Other approaches, such as open source foundations and Government-developed software, offer hope in dealing with some of these issues, but often come with their own problems such as lack of support and version control.

4.3 Extensibility

Network simulators must be able to support a wide variety of physical components that form the network. Some models, such as 802.11 that have wide usage, form the backbone of most network simulators. However, new devices, protocols, and routing configurations require network simulators that can accommodate these new proposed systems. Adding these new extended models to a network simulator should be relatively straightforward and parallelizable. Sufficient documentation should be available to facilitate this process.

4.4 Complexity

Complexity can be hard to measure, and there is no lack of material on the subject in the literature. We identify two that we consider most important. One is the source lines of code metric. If two functions are available to compute the same result and one is of much higher cardinality in terms of lines of code (assuming the same computer programming language), then the one with the minimal number of lines would likely be the preferred method. Complexity has a complex interaction with overall performance; however, it must be balanced against that metric. For example, some language semantics will allow for a one line instruction that actually has several complex operations contained within. Using these approaches will technically generate a code that has fewer source lines of code, but can be much more complex in terms of readability and understanding.

The other measurement comes from concepts dealing with code coverage. Cyclomatic complexity analyzes the control flow through software (*16*). The higher the cyclomatic complexity, the more decision points and branches there are through the code. We chose this metric mainly from the consideration of code modifications and extensions. Codes with high cyclomatic complexity can be difficult to fully understand and modify as the conditionals become overwhelming to track.

Assessing the complexity of different PDES methodologies is also problematic. For example, conservative PDES systems do not have to worry about unrolling network events since they do not have the potential for causality error. Optimistic simulators have the added burden of unrolling out-of-order events. There are ongoing efforts to make this process easier through the use of a reverse computation technique (*17*). Here, rollback is achieved by inverting the original operations of the PDES. This approach has the potential to be automated through source code analysis. Once again, the level of complexity here must be balanced against the overall performance achieved.

4.5 Ease of Installation

Building, porting, and compiling software can quickly become an onerous task. Installation on scalable HPC systems requires the additional burden of linking in parallel execution libraries and/or source code. Ideally, software should be built upon industry standard methodologies for code unpacking, compiling, and installation. Overly complicated build packages and systems hinder productivity and often introduce the need for more software so that the build systems themselves will work.

4.6 Maintenance and Support

Software support always comes with a price. Sometimes this price is directly paid to vendors of software products at varying degrees. This could be support for new releases, access to technical support via Web or phone, or some combination of the two. Sometimes this support, especially with open source efforts, comes more in the form of extra effort from the software users to provide services for which they are not paying. This could be longer build times in unfamiliar operating system configurations, tracking down bugs that might be common across the software release or only pertinent to changes made in validated source code, or providing new functionality to push the software forward in capabilities.

4.7 Verification and Validation

Network simulators and the PDES systems they are built upon must be trusted for accuracy. Validation usually means that we are solving the right problem. Simulation results are compared against analytical solutions or empirical data gathered through experimentation. For verification, and especially in the world of parallel computing, we want to make sure that our results do not vary due to using more than one processor. Results should be accurate and consistent across a wide range of processor counts. Ideally, network simulators will contain some set of cases where builds and installations can be tested for accuracy.

5. Overall Assessment

In the course of conducting research and development under the auspices of the MNMI, we have had the chance to analyze and appreciate the state of the art in network emulation, experimentation, and simulation. The field is complex in that there are numerous ways devices can be networked, interact with their physical environment, and be fine-tuned with competing constraints and optimization parameters. Because of this, we have observed a tendency for isolated pockets of research to form within these various disciplines—not a good thing for overall MANET discovery and optimization.

We have outlined what we consider to be the primary and secondary factors when it comes to efficient and meaningful scalable network simulations. It is our opinion that these elements should be considered during any scalable, parallel network simulation tool. These metrics can also be used to assess current and developing approaches. In fact, in the end, many factors come into play without a clear-cut equation to select an optimal approach. We hope that this discussion at least helps.

6. Conclusions

Network simulation is one of the key enabling technologies to allow for optimal network planning, assessment, and optimization. With the increasing digitization of the Army and network-centric warfare, being able to research and investigate the various routing algorithms, physical effects, and device interaction has never been more important.

Various metrics will determine how well network simulation tools perform when scaling to numbers commensurate with current and future Army deployments of MANETs. These consist of a mix of ease of use and integration capabilities, ability to accurately model real-world conditions, problem setup and representation, and scaling and performance across a range of HPC processors including traditional multi-core von Neumann processors and streaming processors such as those found in GPUs. By achieving success in each of these areas, network simulators can be an effective component in Army MANET optimization and planning to ensure mission success.

References

[1] Clarke, J.; Renard, K. NetDMF: A Scalable Active Data Hub for Mobile Network Modeling. *HPCMP Users Group Conference* **2009**, *0*, 321–325.

[2] Szymanski, B.; Madnani, K. Wireless/Mobile Network Simulation Under Genesis. In *DARPA NMS PI Meeting*, 2002.

[3] Clinton Kelly, I. Wireless Network Simulation Done Faster than Real Time, Master's thesis, Cornell University, 2003.

[4] Khosroshahy, M. Study and Implementation of IEEE 802.11 Physical Layer Model in YANS (Future NS-3) Network Simulator, Master's thesis, Ecole Nationale Supérieure des Télécommunications, 2006.

[5] ns-3 website, http://www.nsnam.org, 2011 Network Simulator-3 (ns-3).

[6] Chang, X. Network Simulations with OPNET. In *Winter Simulation Conference*, 1999.

[7] Khosa, I.; Haider, U.; Mosood, H. Evaluating the Performance of IEEE 802.11 MAC Protocol using OPNET Modeler. In *2010 International Conference on Electronics and Information Engineering (ICEIE 2010)*, 2010.

[8] Bauer, D.; Yaun, G.; Carothers, C.; Yuksel, M.; Kalyanaraman, S. ROSS.Net: Optimistic Parallel Simulation Framework for Large-Scale Internet Models. In *Winter Simulation Conference*, 2003.

[9] Fujimoto, R. M. Parallel Discrete Event Simulation. *Commun. ACM* **1990**, *33*, 30–53.

[10] Perumalla, K. S. Scaling Time Warp-based Discrete Event Execution to 104 Processors on a Blue Gene Supercomputer. In *Proceedings of the 4th international conference on Computing frontiers*; CF '07, ACM: New York, NY, USA, 2007.

[11] Carothers, C. D.; Bauer, D.; Pearce, S. ROSS: A High-performance, Low Memory, Modular Time Warp System. In *Proceedings of the fourteenth workshop on Parallel and distributed simulation*; PADS '00, IEEE Computer Society: Washington, DC, USA, 2000.

[12] Weingartner, E.; vom Lehn, H.; Wehrle, K. A Performance Comparison of Recent Network Simulators. In *Communications, 2009. ICC '09. IEEE International Conference on*; 2009.

[13] Bader, D. A.; Moret, B.M.E.; Sanders, P. Algorithm Engineering for Parallel Computation. In Springer-Verlag New York, Inc.: New York, NY, USA, 2002.

[14] Henz, B. J.; Shires, D. R. Parallel Finite Element Software Development and Performance Analysis in an Object-Oriented Programming Framework. *Journal of Mathematical Modelling and Algorithms* **2005** , *4*, 17–34; 10.1007/s10852-004-3520-4.

[15] Ukrainsky, O.; Zebrowitz, H.; Hein, C.; Cortese, A.; Rubin, A.; Poon, C.; Bard, A.; Reyes, H. An Open Environment for Rapid Embedded Planning of On-the-Move Communications Networks Using Multi-level Abstraction. In *Military Communications Conference, 2005. MILCOM 2005. IEEE*, 2005.

[16] McCabe, T. J. A Complexity Measure. *IEEE Transactions on Software Engineering* **1976**, *SE-2 (4).*

[17] Carothers, C. D.; Perumalla, K. S.; Fujimoto, R. M. Efficient Optimistic Parallel Simulations Using Reverse Computation. *ACM Trans. Model. Comput. Simul.* **1999**, *9*, 224–253.

List of Symbols, Abbreviations, and Acronyms

ARL	U.S. Army Research Laboratory
COMPOSER	Communications Planner for Operational and Simulation Effects with Realism
DES	discrete event simulator
DoD	Department of Defense
GPUs	graphics processing units
GUIs	graphical user interfaces
HPC	high performance computing
IP	internet protocol
MANETs	mobile *ad hoc* networks
MNMI	Mobile Network Modeling Institute
NetDMF	Network Data Model and Format
NICE	Network Interdisciplinary Computing Environment
OSI model	Open Systems Interconnection model
PDES	parallel discrete event simulation
TLB	translation lookaside buffer
UDP	User Datagram Protocol
UGS	unattended ground sensors
YANS	Yet Another Network Simulator

NO. OF	
COPIES	ORGANIZATION
1 ELEC	ADMNSTR DEFNS TECHL INFO CTR ATTN DTIC OCP 8725 JOHN J KINGMAN RD STE 0944 FT BELVOIR VA 22060-6218
1	RENSSELAER POLYTECHNIC INSTITUTE ATTN C CAROTHERS 110 8TH STREET TROY NY 12180
1	C4ISR & NETWORK MODERNIZATION ATTN RDER OT D KU BLDG 6007 COMBAT DRIVE F2-101 ABERDEEN PROVING GROUND MD 21005
4	US ARMY RSRCH LAB ATTN RDRL CIH C B HENZ ATTN RDRL CIH C D SHIRES ATTN RDRL CIH C J CLARKE ATTN RDRL CIH C K RENARD ABERDEEN PROVING GROUND MD 21005
1	US ARMY RSRCH LAB ATTN RDRL CIH R NAMBURU ABERDEEN PROVING GROUND MD 21005
3	US ARMY RSRCH LAB ATTN IMNE ALC HRR MAIL & RECORDS MGMT ATTN RDRL CIO LL TECHL LIB ATTN RDRL CIO MT TECHL PUB ADELPHI MD 20783-1197